Machine Learning Engineer Nanodegree

Capstone Project

Sonja Tilly May 2017

I. Definition

Project Overview

Lending Club is the world's largest peer-to-peer online platform, which has enjoyed tremendous success since its foundation in 2007.
Borrowers are able to apply for loans, which are rated – the lower the rating, the higher the interest rate. Investors can purchase notes of those loans. As borrowers make payments on their loans, investors are paid monthly on their respective notes, less the impact of any defaults and Lending Club fees. According to Lending Club, the estimated annual net return on a diversified loans portfolio is 5%[1], which remains an attractive return in the current low interest rate environment, with the 5Y US Treasury yield below 2%.

A number of research pieces have been published by various bodies, each examining different aspects of the dataset.
Research published on the NY Data Science Academy website segments the Lending Club dataset 'good' and 'bad' loans and applies a logistic regression model to estimate the expected loss on loans listed as 'current' [2]. BlackMoon Financial Group analysed non-numerical data and demonstrated how this information can help investors make better decisions when manually selecting loans for their portfolio[3]. Researchers from Stanford University used supervised learning techniques to predict the probability that a loan application will get approved. Further, they used various visualization techniques to show patterns in the dataset[4]. A paper by a student at General Assembly determined the key features indicative of loan defaults and built models to predict them[5].

Working in Asset Management with focus on fixed income strategies, I found this dataset particularly interesting as it may hold deeper insights into the dynamics of a relatively new yet fast-growing market segment and how investors could enhance their asset allocation when manually constructing a portfolio of loans and ultimately achieve a better return on their investment.

The dataset obtained from Kaggle[6] contains complete loan data for all loans issued between 2007 and 2015, including the loan status (current, late, fully paid, etc.) and payment information. The data contains a range of different data types such as numerical data, free form text and date objects, which means that the raw data will require pre-processing. The file is a table of 887,379 observations and 74 variables. A data dictionary is provided in a separate file. The dataset will be used to examine the relationships between loan performance and other variables.

1 https://www.lendingclub.com/investing/alternative-asset-investments/how-it-works

2 http://blog.nycdatascience.com/r/p2p-loan-data-analysis-using-lending-club-data

3 https://blackmoonfg.com/site/pdf/bm_non_numerical_data.pdf

4 http://Pcs229.stanford.edu/proj2016spr/report/039.pdf

5 https://res.cloudinary.com/general-assembly-profiles/image/upload/v1416535475/uwumooopptsmpgu1goo.pdf

6 https://www.kaggle.com/wendykan/lending-club-loan-data

Machine Learning Engineer Nanodegree

Capstone Project

Sonja Tilly May 2017

Problem Statement

Lending Club allows investors to purchase notes in loans based on their credit rating. However, investors have no means of predicting the timeliness of cash flows of any one loan. Having this information could significantly enhance investors' asset allocation strategy as they would be able to avoid those loans where delayed or wiped out cash flows are expected given the same interest rate level. "Avoiding the losers" is a predominant performance driver in debt investing and hence, the key concern for any Lending Club investor is the timeliness of cash flows.

The goal of this project is to predict if any one loan in the dataset will generate timely cash flows or will suffer from delayed/impaired cash flows. This is a binary classification problem where zero is a "bad" loan and one is a "good" loan. Accordingly, loans have been assigned classifications according to their status:

- "Current" = 1
- "Issued" = 1
- "Fully Paid" = 1
- "Charged Off" = 0
- "Late" = 0
- "In Grace Period" = 0
- "Default" = 0

The above classification represents the target variable on which predictions will be made.
In this paper, any loan classified as a "1" will be referred to as "good loan" while any loan classified with a "0" will be referred to as "bad loan".

The approach to tackling this project includes several stages:

Data visualization. At this stage, the relationships between different variables and loan performance will be explored visually. Preliminary findings will be discussed and summarized.

Data pre-processing. The Lending Club dataset contains various data types such as numerical data, or date objects. Some variables such as loan purpose or job title are in text format and have to be converted.
Further pre-processing required for modelling could include transforming categorical variables and dealing with null values.

Feature selection. Insights from the above may be helpful when selecting features that are relevant to predicting loan performance.

Modelling. A logistic regression model will be built as benchmark model and performance will be assessed using the AUC score.
Different models will be constructed using algorithms such as Naïve Bayes, RandomForests and Neural Networks. All three algorithms are suited to a classification problem. The dataset includes a large number of observations, particularly required by the RandomForest and Neural Network algorithms. Model performance will be assessed using the AUC score compared to that of the benchmark model.
Model optimization methods such as GridSearch will be applied to improve performance.
Optimized model performance (including benchmark model) will be compared to the original models and the benchmark model.

Metrics

The dataset is unbalanced, with bad loans accounting for only a small portion of the total number of loans. Due to this class imbalance, model accuracy will be assessed the Area Under the Curve (AUC).
A receiver operating characteristic curve or ROC curve is a graph plotting the true positive rate against the false positive rate at different thresholds. It indicates how well a classifier can discriminate positive and negative instances and identify the best threshold for discriminating them. The curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.

- TPR (also called Recall or Sensitivity) = sum total positives/sum of condition positive
- FPR (also called Fall-out) = sum of false positives/sum of condition negative

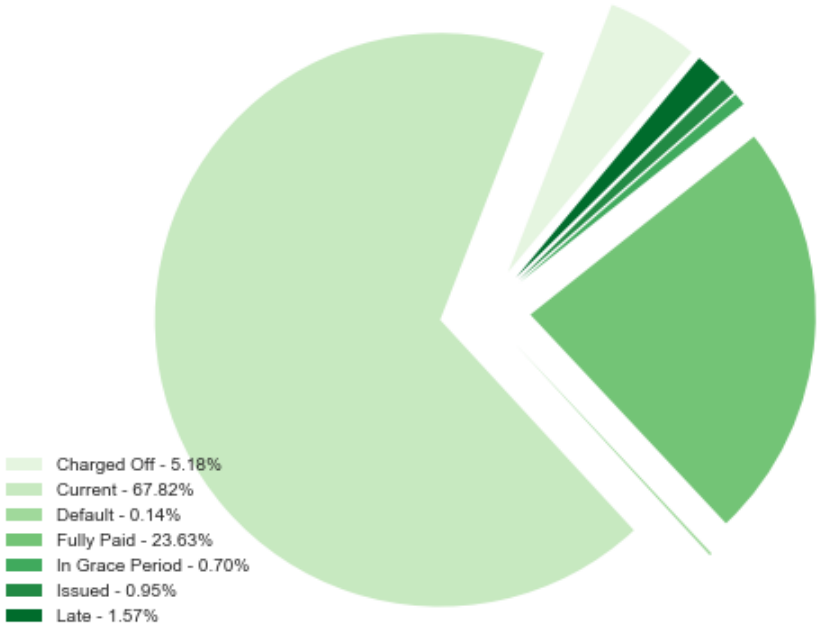
The area under the curve (AUC) is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one[7].

II. Analysis

Data Exploration

The Lending Club dataset has 887,379 entries and 74 data columns describing a wide range of loan attributes such as loan grade, interest rate, loan status, purpose or zip code of loan applicant. It contains a mix datetime objects (1), floats (49), integers (3) and object (23) data types.
Pre-processing is required to convert non-numerical variables into numerical ones for modelling. Further pre-processing required includes transforming categorical variables and dealing with null values. Some features do not have predictive power and can be eliminated for modelling purposes.

Figure 1



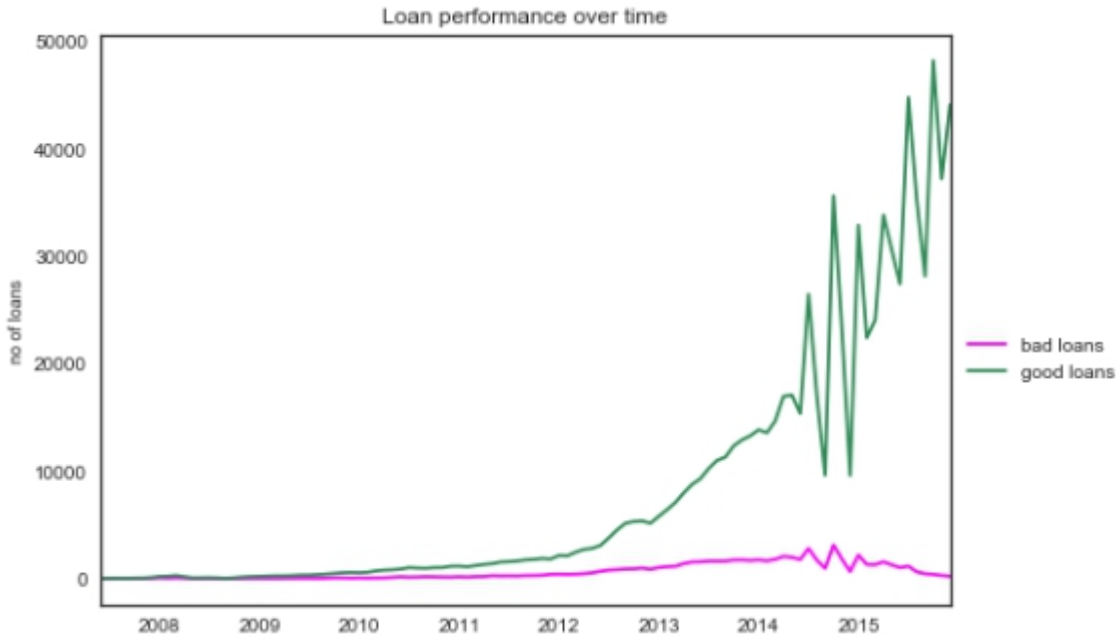
7 www.wikipedia.com

The dataset is highly unbalanced, with good loans (=Current, Fully Paid and Issued) accounting for 92.4% of all loans.

Exploratory Visualization

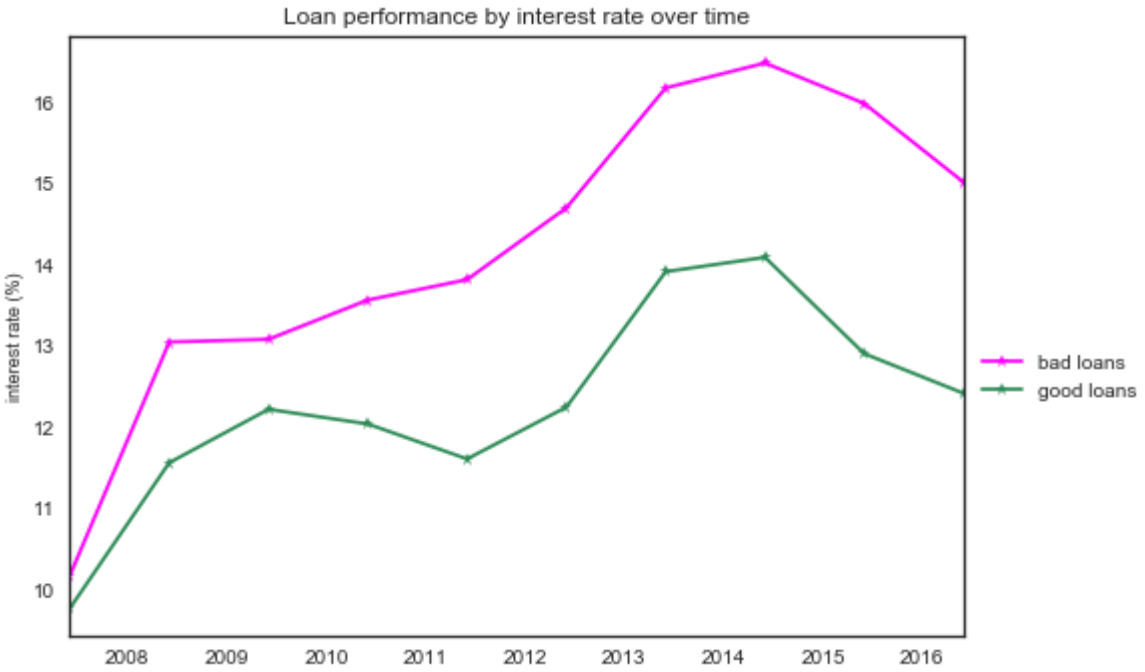
In this section, I will visually explore the dataset to gain deeper insights into the relationships of different variables with loan performance.

Figure 2: Loan performance over time



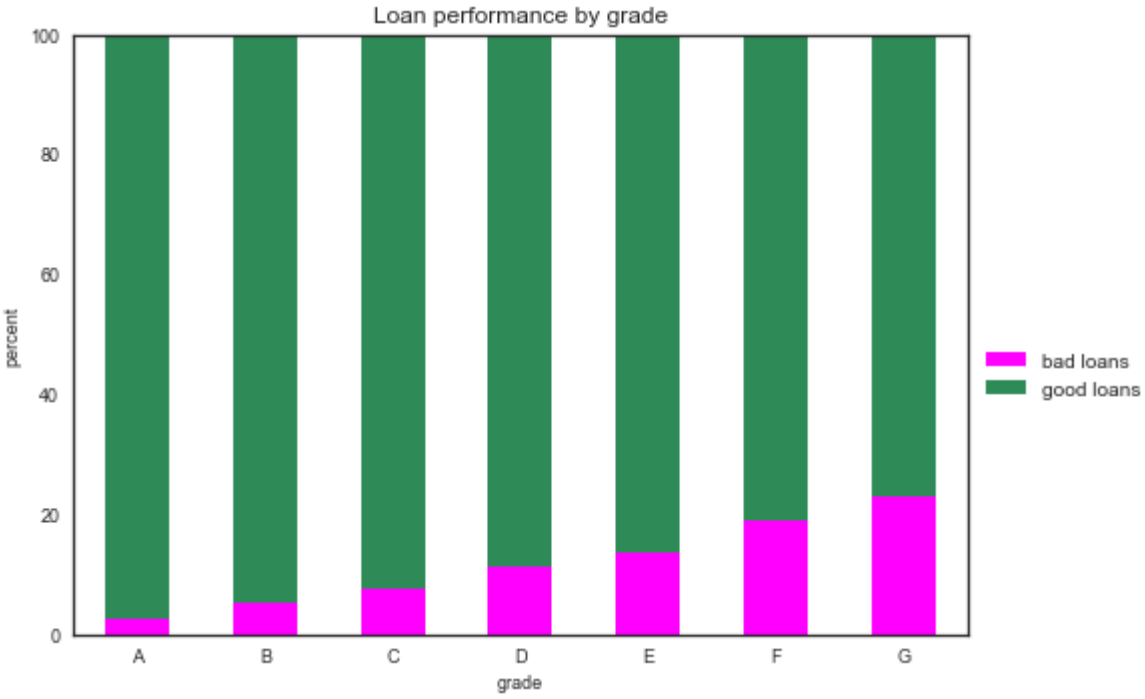
Since its foundation in 2007, Lending Club has enjoyed a significant success, which is reflected in a strong increase in the volume of loans issued. Interestingly, the volume of bad loans has peaked around 2014 and then started to decline. This could be due to stricter lending standards as bad loans are tend to be concentrated in the lower quality segments.

Figure 3: How is loan performance related to interest rates?



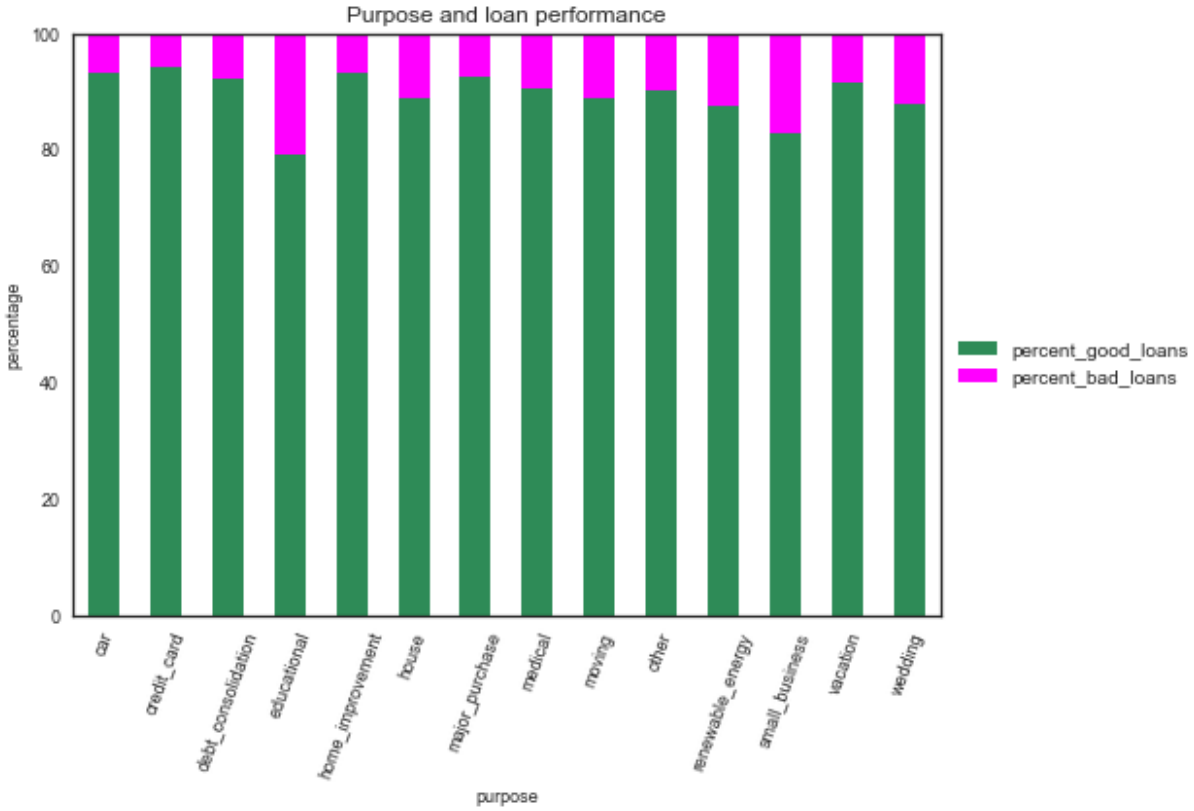
Good loans seem to be, on average, characterised by lower interest rates while the opposite is true for bad loans. It appears that the interest rates for both good and bad loans have peaked in 2014 and have declined since. This can be explained by a tightening in lending standards, resulting in less issuance of loans at higher interest rates (i.e. at the lower end of the grade spectrum).

Figure 4: What is the relationship between loan performance and grade?



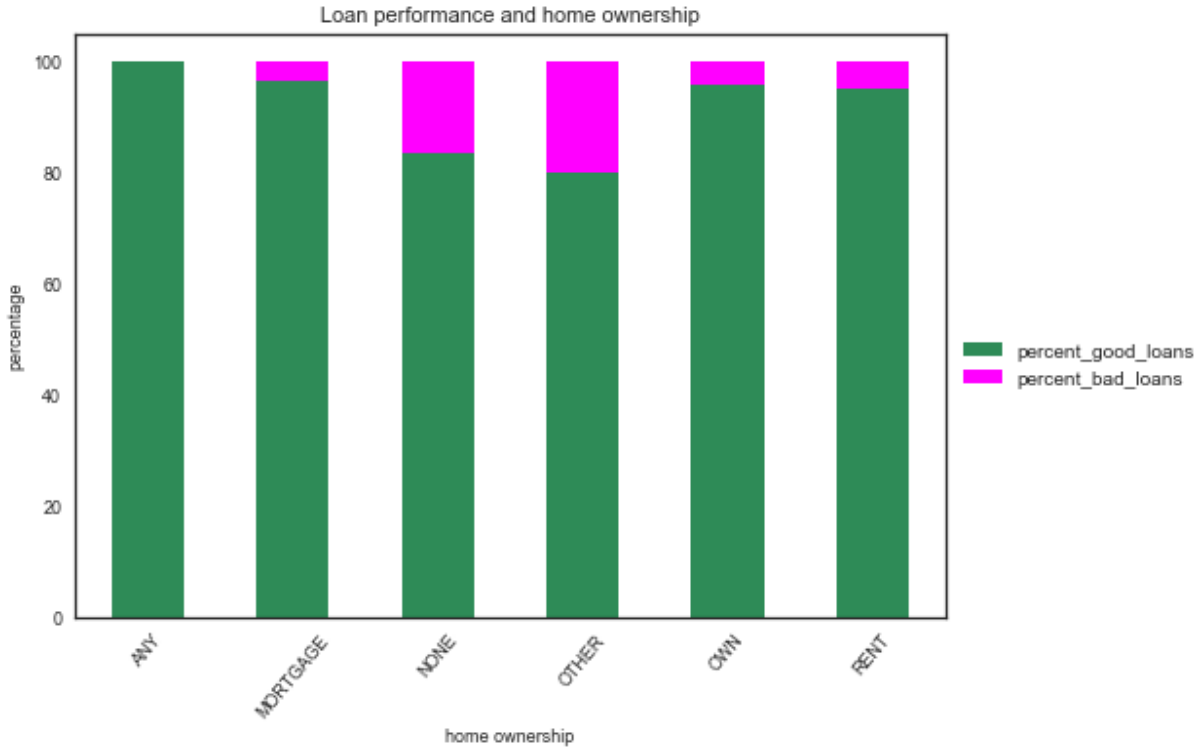
The portion of bad loans increases in proportion to declining loan quality, which is intuitive.

Figure 5: What is the relationship between loan performance and purpose?



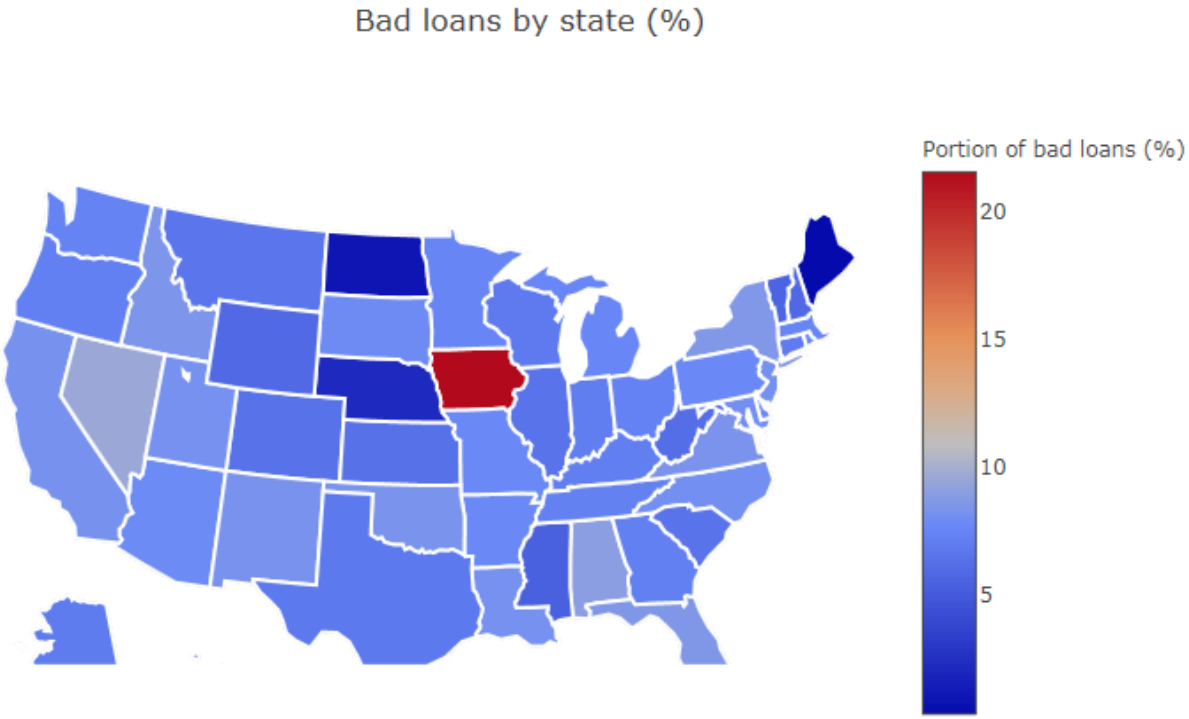
Interestingly, the portion of bad loans intended for education or small businesses is visibly higher than other purposes. Loans taken to cover credit card debt have the lowest portion of bad loans, followed by home improvement and car loans.

Figure 6: What is the relationship between loan performance and home ownership?



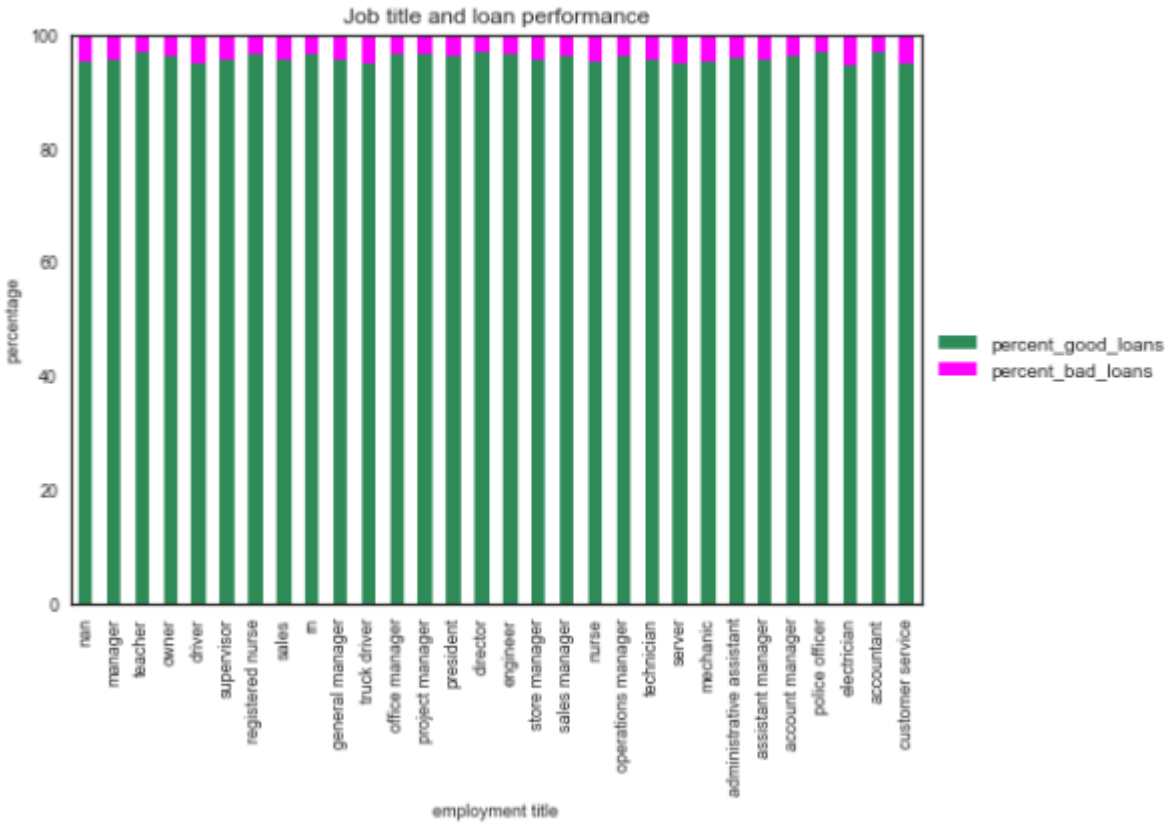
Bad loans are concentrated where applicants have not indicated any home ownership i.e. 'None' or 'Other'.

Figure 7: What is the relationship between loan performance and applicant state?



Bad loans are concentrated in Iowa with 21.43%. However, only 14 loans have been made to applicants from Iowa, a tiny proportion of the entire dataset. The largest number of loans (129,517) has been issued to applicants from California. The percentage of bad loans in California is only 0.91% over the mean percentage of bad loans per state (7.21%).

Figure 8: What is the relationship between loan performance and employment title?



Blue-collar jobs such as driver, truck driver, server or electrician seem to have a marginally higher percentage of bad loans than so-called white collar jobs. However, I feel that this column has a minimal predictive power due to the large number of job titles (only top 20 shown above) and hence this variable has been taken out.

Algorithms and Techniques

Models will be constructed using the following algorithms:

- **Naïve Bayes**. It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. A Naive Bayes classifier assumes that a particular feature in a class is independent any other feature. The Naive Bayes model is relatively simple and tends to perform well with very large data sets. One of the main limitations of Naive Bayes is the assumption of independent predictors, which is unrealistic[8].
- **Multilayer Neural Perceptron.** The building blocks for neural networks are artificial neurons. These are simple computational units that have weighted input signals and produce an output signal using an activation function. Neural networks have the ability to learn the representation in the training data and how to best relate it to the target variable. The predictive capability of neural networks comes from the multi-layered structure of the networks. The data structure can pick out features at different scales or resolutions and combine them into higher-order features[9].

8 https://www.analyticsvidhya.com/blog/2015/09/naive-bayes-explained/

- **Ensemble methods such as RandomForests or XGBoost.**

  The main principle behind ensemble methods is that a group of "weak learners" can come together to form a "strong learner".

  The random forest starts with a standard machine learning technique called a "decision tree" which, in ensemble terms, corresponds to a weak learner. In a decision tree, an input is entered at the top and as it traverses down the tree the data gets bucketed into smaller and smaller sets[10].

  XGBoost is an implementation of gradient boosted decision trees designed for speed and performance. Gradient boosting is an approach where new models are created that predict the residuals or errors of prior models and then added together to make the final prediction. It is called gradient boosting because it uses a gradient descent algorithm to minimize the loss when adding new models[11].

All above algorithms are suited to classification problems involving a large amount of data.
The models will be trained using the training set.
Model performance will be assessed using the AUC score calculated on the test set compared to that of the benchmark model.
Model optimization methods such as GridSearch and stacking will be applied to improve performance.
The predictions from the Naïve Bayes, Multi-Layer Perceptron, Regression and RandomForest models will be added as additional features to the training and testing datasets. An XGBoost classifier will be trained and test on the enhanced dataset.
A validation set has been split out from the test set. Model performance of the models will be tested on the validation set to avoid overfitting to the test set.
Final model performance will be compared to that of the benchmark model.

Benchmark

The benchmark for this project is a simple logistic regression model, appropriate as the project represents a classification problem. The model is used to predict the probability of a "good loan" or a "bad loan" as defined in the problem statement. Benchmark model performance is measured by the AUC score.

III. Methodology

Data Pre-processing

In the pre-processing stage, the following points have been addressed:

- **Deal with NAN values**. NAN values were replaced by "0".
- **Drop variables that are not required.** Variables dropped were:
  - 'id', 'member_id', 'policy_code', 'url' - these variables contain no predictive power;
  - 'desc', 'title' - the variable 'purpose' contains the same keywords also included in the latter two (see wordclouds in 'Capstone Project NB1');
  - 'collections_recovery_fee', 'collections_12_mths_ex_med', 'total_rec_late_fee', 'recoveries', 'acc_now_delinq', 'pymnt_plan' – these variables relate to collections on defaulted loans;

9 http://machinelearningmastery.com/neural-networks-crash-course/

10 http://blog.citizennet.com/blog/2012/11/10/random-forests-ensembles-and-performance-metrics

11 http://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/

- 'funded_amnt', 'funded_amnt_inv', 'initial_list_status' - these relate to the funded status of a loan after issuance;
- 'emp_title' – from the EDA, we have learnt that the variable seems to have minimal predictive power and has thus been taken out.
- 'verification_status', 'verification_status_joint' – this relates to income verification however I hold the view that this is a driver of funding rather than performance; hence these features have been taken out.
- 'loan_status', 'loan_status_new' – these columns have been converted into the target variable.

- **Encode variables that are objects.** The columns 'term', 'grade' 'sub_grade', 'home_ownership', 'verification_status', 'verification-status_joint', 'pymnt_plan', 'purpose', 'zip_code', 'addr_state', 'initial_list_status', 'application_type', 'emp_length', 'emp_title' have been encoded in order to convert non-numerical values into numerical ones.

- **Convert date objects into ordinal numbers.** The columns 'issue_d', 'earliest_cr_line', 'last_pymnt_d', 'next_pymnt_d', 'last_credit_pull_d' have been converted into ordinal numbers have been converted from datetime objects to ordinal numbers.

Implementation

- A train/test split was implemented on the dataset.

- The following models were built and the respective classifiers were trained on the pre-processed training data. The AUC score was calculated on each using the test data:

Classifier	AUC score
Logistic regression (benchmark)	0.8206
Gaussian Naive Bayes	0.6095
Multi-Layer Perceptron	0.5000
RandomForests	0.9107

Refinement

Gridsearch was conducted to tune the hyperparameters of each model. This is not applicable to the Gaussian Naïve Bayes classifier, which does not have hyperparameters.

Classifier	Improved AUC score	Parameters tuned
Logistic regression (benchmark)	0.8383	'C' (= inverse of regularisation strength); 'class_weight'
Multi-Layer Perceptron	0.7822	'hidden_layer_sizes', 'activation', 'alpha', 'learning_rate_init'
RandomForests	0.9242	'n_estimators', 'max_depth', 'min_samples_split', 'class_weight', 'n_jobs'

- For the logistic regression model, tuning the 'class_weight' parameter had the most impact as the classes are very unbalanced.

- For the Multi-Layer Perceptron model, tuning 'hidden_layer_sizes' and 'activation' had the most impact.
- For the RandomForests model, adjusting 'class_weight' and 'max_depth had the most impact on performance.

The optimised RandomForests model delivered the highest AUC score, 0.9242. While the score is good, other avenues should be explored to further improve performance.

Therefore, the results of the optimised models were added as additional features to the train and test datasets. The score generated by the Naive Bayes model was regarded too low, hence this model was not used.

An XGBoost model was built and trained on the 'enhanced' training dataset. Parameter tuning was conducted and the following combinations delivered the best score on the test set:

```
param = {
        'task': 'train',
        'boosting_type': 'gbdt',
        'objective': 'binary', # binary if you have one class to predict
        'num_classes': 1,
        'metric': 'auc', #there is also AUC
        'max_depth': 10,
        'learning_rate': 0.01,
        'feature_fraction': 0.8,
        'bagging_fraction': 0.8,
        'num_thread':1,
        'bagging_freq': 5,
        'verbose': 0,
        'num_iterations':80, # number of features
        'is_unbalanced':True,
        'early_stopping_round':5 # stops if no improvement after 5 iterations
        }

num_round = 2000 # number of iterations

lgb.cv(param, train_data, num_round, nfold=5) # it cross validates with 5 folds for selection
bst = lgb.train(param, train_data, num_round, valid_sets=[test_data] )
```

IV. Results

Model Evaluation and Validation

The XGBoost model was chosen as the final model. As anticipated, the model performed strongly, generating an AUC score of 0.9841.

XGBoost is an implementation of gradient boosted decision trees designed for speed and performance[12]. The package is widely used and known for its good results, for instance in Kaggle competitions.

The final parameters appear appropriate, most of which have been found by trial and error. Setting parameter *'is_unbalanced'* to 'True' had a particularly positive impact on performance as the dataset is

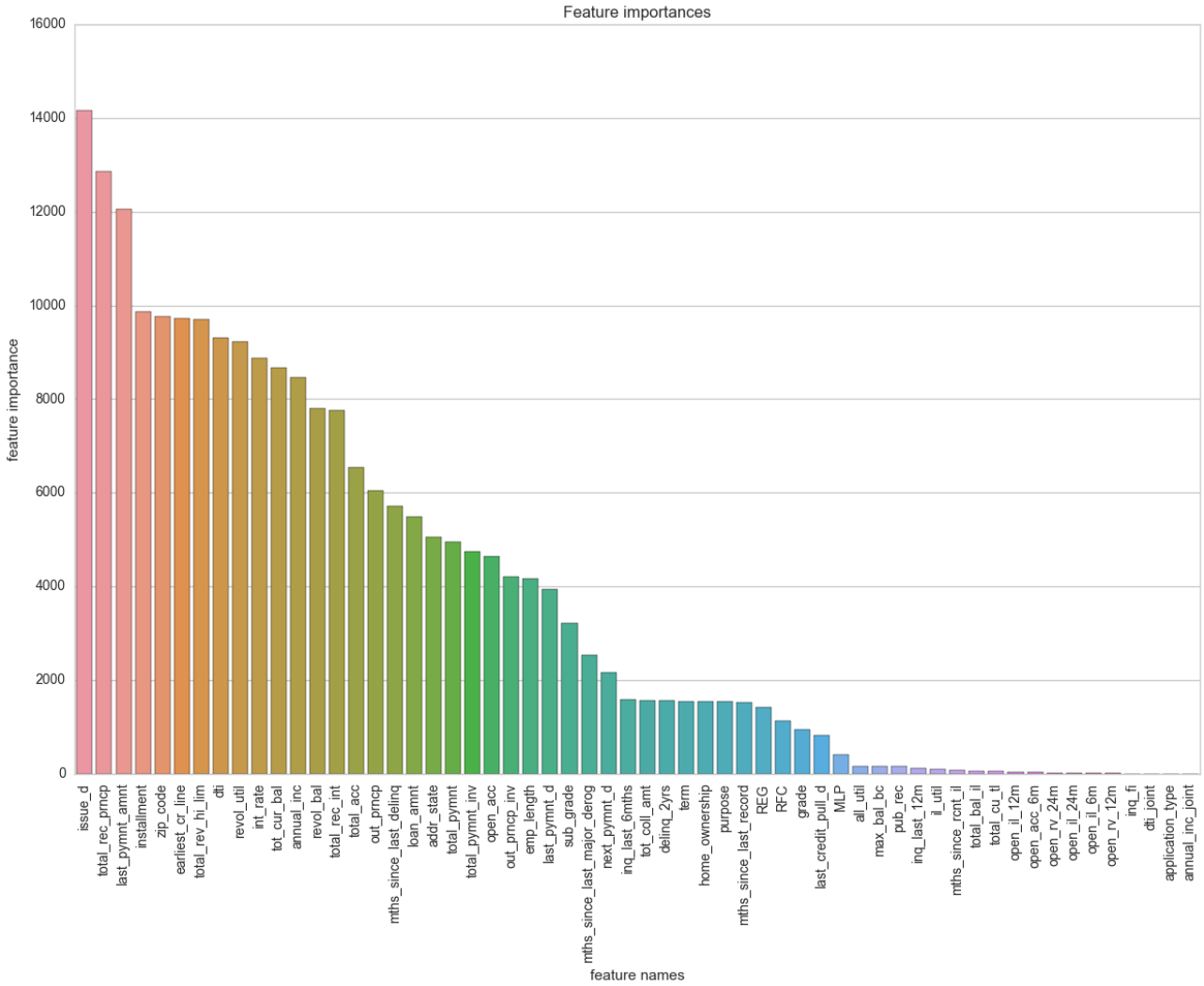
12 http://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/

highly unbalanced. Further, increasing the number of iterations to 2,000 impacted performance meaningfully.

The model ran a 5-fold cross-validation on the test data. The AUC score indicates that the model generalises well to unseen data.

Examining the feature importance, it becomes apparent that there is a 'tail' of features with insignificant predictive power:

Figure 9: Feature importance

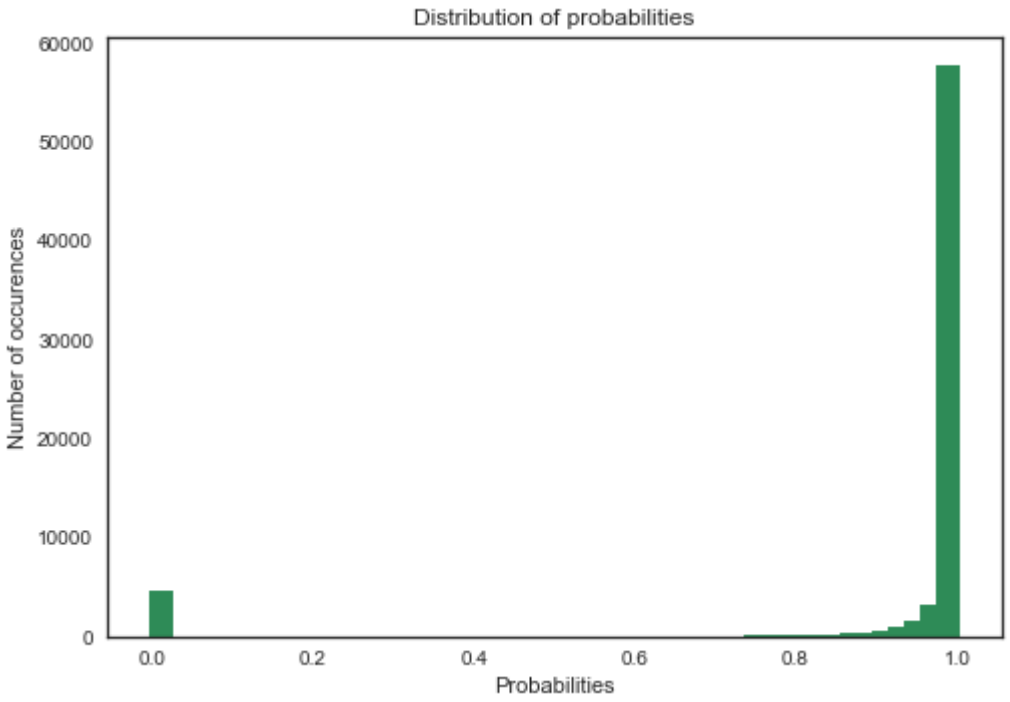


In terms of sensitivity analysis, the three features derived from the output of the optimised models were removed from the dataset. The model was re-run using the altered dataset, with the model generating an AUC score of 0.9855. This indicates that small changes in the training data do not greatly affect the results and that the model is therefore robust.

Justification

The XGBoost model achieved an AUC score of 0.9841 on the test data, beating the basic benchmark model, which has delivered 0.8206, by a generous margin of 0.1635.

Figure 10: Distribution of prediction probabilities



The predicted probabilities are mostly unambiguous. Of 887,379 entries, 105 have predicted probabilities between 0.4 and 0.6, which can be considered ambiguous (0.01% of the entire dataset).

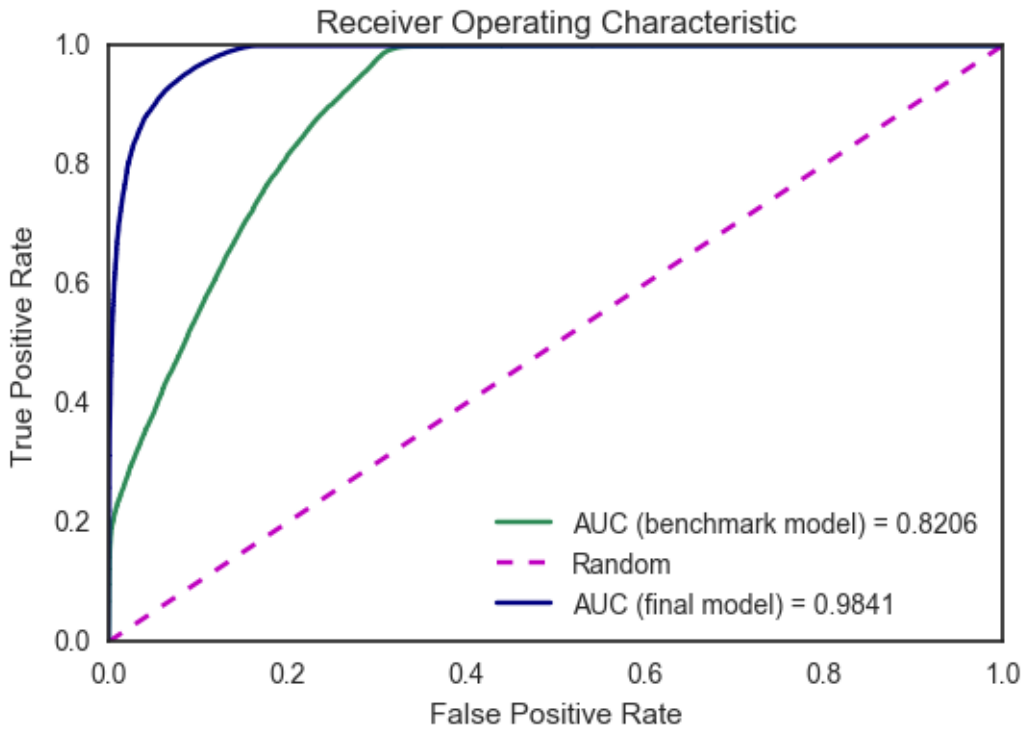
Both the score and the distribution of predicted probabilities indicate that the model's results are sufficiently significant to solve the problem posed in this project.

V. Conclusion

The below chart shows the Receiver Operating Characteristic (ROC) curves for a random approach, the benchmark model and the final model. A ROC curve illustrates the diagnostic ability of a binary classifier as its discrimination threshold varies. The ROC curve plots the true positive rates against the false positive rate ("false alarm"). The below chart shows the ROC curves for random guessing, the benchmark model and the final model. A "perfect" classifier would generate coordinate (0, 1) of the ROC space, showing no false negatives and no false positives[13].

13 https://en.wikipedia.org/wiki/Receiver_operating_characteristic

Figure 10: Receiver Operating Characteristic curve



The ROC curve of the final model indicates superior performance to the benchmark model.

Reflection

This project can be summarised in the following steps:

1. Definition of the problem using a publicly available dataset
2. Exploratory data analysis
3. Pre-processing of the dataset and feature selection
4. Creation of the benchmark model and performance assessment
5. Creation of a range of models and performance assessment
6. Model improvement and performance assessment
7. Addition of the respective model predictions on test data as new features in the dataset
8. Creation of final model , parameter tuning and performance assessment
9. Sensitivity testing on final model
10. Make predictions using the validation set on final model and performance assessment

Parameter tuning was particularly challenging and a variety of combinations had to be tried prior to achieving improved results.

Further, XGBoost had not been used before and required familiarisation. The speed and performance of this software is impressive and it will certainly be considered to solve this type of problem in future projects.

A software that could be used to solve classification problems is Keras, which runs on Tenserflow. However, Keras had not been used before and training would be required in order to use the software effectively.

Improvement

In this project, a range of models was built, optimised and added their predictions as additional features to the dataset. The sensitivity test and an analysis of the feature importance suggested that there was no significant added value was derived from these additional features.

- To simplify the project structure, only build one model could be built using XGBoost.
- The XGBoost model's parameters could be further tuned to achieve an even better score. Further, a higher numbers of iterations could be explored.
- New features could be engineered for instance ratios such as monthly instalment to monthly income.
- NAN values have been dealt with by filling them with zero. However, a better way may be to fill NAN values with the feature's mean or median value.